L'Attaque de Coppersmith sur le Bourrage Court (Short Pad Attack)

Dylan Mona, Adya Sarr

October 28, 2025

Introduction et Contexte de l'Attaque

Contexte

L'attaque a Bourrage Court de Coppersmith(Short Pad Attack) cible le chiffrement **RSA**.

L'Algorithme LLL

Lenstra–Lenstra–Lovász (LLL)

- Soit $L \subset \mathbf{Z}^n$ un réseau engendré par une base $B = \{b_1, \ldots, b_n\}$.
- L'algorithme LLL produit un vecteur $v \in L$ satisfaisant :

$$\|v\| \leq 2^{\frac{n-1}{4}} \left(\det(L)\right)^{1/n}$$

- Cet algorithme s'exécute en temps polynomial.

Lemme de Håstad/Howgrave-Graham

Énoncé

Soit $g(x) = \sum_{i=0}^{n} c_i x^i$ un **polynôme uni-varié à coefficients entiers** X et m deux **entier positif** tel que:

1. $g(x_0) \equiv 0 \mod N^m$ où $|x_0| \leq X$
2. $||g(xX)|| < \frac{N^m}{\sqrt{n}}$ où n est le nombre de monômes dans $g(x)$

Alors $\mathbf{x_0}$ est une **racine** sur les **entiers** $(\mathbf{Z})$ c'est-a-dire $\mathbf{g}(\mathbf{x_0}) = 0$

Preuve

$|g(x_0)| = \left|\sum_i c_i x_0^i\right| \leq \sum_i |c_i \cdot X^i \frac{x_0^i}{X^i}| \leq \sum_i |c_i \cdot X^i| \leq \sum_i |c_i| X^i \leq \sqrt{n} \cdot ||g(xX)||_{L_2}$.

La dernière inégalité est la version pour L_2 de l'inégalité de Schwarz.

En appliquant la condition 2. on a:

$$|g(x_0)| < \sqrt{n} \cdot \frac{N^m}{\sqrt{n}} = N^m$$

On a $g(x_0) \equiv 0 \mod N^m$(condition 1) donc $g(x_0)$ est un **multiple de** N^m.

Comme $|g(x_0)| < N^m$ alors $g(x_0) = 0$

Théorème de Coppersmith

Énoncé du Théorème

- Soit
$$N \in \mathbf{Z}, \quad f(x) \in \mathbf{Z}[x]$$
un polynôme unitaire de degré d.
- Posons
$$X = N^{(1/d)}.$$
- Alors, il existe un algorithme qui, étant donnée la paire (N, f), trouve tous les entiers
$$x_0 \in \mathbf{Z} \quad \text{tels que} \quad |x_0| < X \quad \text{et} \quad f(x_0) \equiv 0 \pmod{N}.$$
- Le temps d'exécution de cet algorithme est polynomial en $\log N$ et est dominé par le coût de l'exécution de l'algorithme LLL.

Preuve du Théorème de Coppersmith (1/4)

Idée générale

- On veut montrer que la recherche d'une **petite racine modulo** N
$$f(x_0) \equiv 0 \pmod{N}$$
peut être reformulée comme la recherche d'un **polynôme à petits coefficients** partageant cette racine.
- Données :
$$N \in \mathbf{Z}, \quad f(x) \in \mathbf{Z}[x] \text{ unitaire de degré } d.$$
- On cherche une petite racine x_0 telle que $|x_0| < X$, avec
$$X = N^{\frac{1}{d}}.$$
- Objectif : construire un **réseau** dont un vecteur court correspond à un polynôme $h(x)$ vérifiant
$$h(x_0) = 0 \quad \text{dans } \mathbf{Z}.$$

Preuve du Théorème de Coppersmith (2/4)

Construction du réseau

- Si $f(x_0) \equiv 0 \pmod{N}$, alors pour tout $k \geq 1$:
$$f(x_0)^k \equiv 0 \pmod{N^k}.$$
- On choisit un entier
$$m \approx \frac{\log N}{d \log X}$$
afin que le réseau soit assez grand pour LLL.
- On définit les polynômes :
$$g_{u,v}(x) = N^{m-v} x^u f(x)^v, \quad 0 \leq u \leq d-1,\ 0 \leq v \leq m.$$
- Ces polynômes satisfont :
$$g_{u,v}(x_0) \equiv 0 \pmod{N^m}.$$

Preuve du Théorème de Coppersmith (3/4)

Mise à l'échelle et matrice associée

- Pour comparer les tailles des coefficients, on effectue une mise à l'échelle :

$$g_{u,v}(x) \mapsto g_{u,v}(xX).$$

- Exemple pour $d = 2$, $m = 3$:

$$g_{u,v}(xX) = N^{m-v}(xX)^u f(xX)^v, \quad 0 \le u \le d-1, \ 0 \le v \le m.$$

$$\mathbf{B} = \left[\begin{array}{c|cccccccc}
 & x^0 & x^1 & x^2 & x^3 & x^4 & x^5 & x^6 & x^7 \\
\hline
g_{0,0}(xX) & N^3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
g_{1,0}(xX) & * & XN^3 & 0 & 0 & 0 & 0 & 0 & 0 \\
g_{0,1}(xX) & * & * & X^2N^2 & 0 & 0 & 0 & 0 & 0 \\
g_{1,1}(xX) & * & * & * & X^3N^2 & 0 & 0 & 0 & 0 \\
g_{0,2}(xX) & * & * & * & * & X^4N & 0 & 0 & 0 \\
g_{1,2}(xX) & * & * & * & * & * & X^5N & 0 & 0 \\
g_{0,3}(xX) & * & * & * & * & * & * & X^6 & 0 \\
g_{1,3}(xX) & * & * & * & * & * & * & * & X^7
\end{array}\right]$$

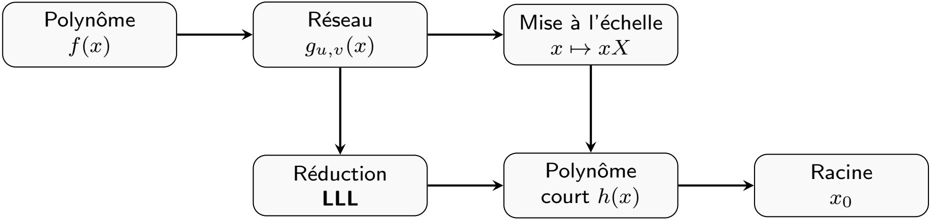
Preuve du Théorème de Coppersmith (4/4)

Application de LLL

- La dimension du réseau est :
$$w = (m+1) \cdot d.$$
- D'après la borne de Hermite, il existe un vecteur non nul $v \in L$ tel que :
$$\|v\| \leq \sqrt{w}\, \det(L)^{1/w}.$$
- En choisissant m suffisamment grand, on assure :
$$\|v\| \leq \frac{N^m}{\sqrt{w}}.$$
- En appliquant LLL, on obtient un polynôme $h(x)$ court vérifiant :
$$\|h(xX)\| \leq \frac{N^m}{\sqrt{w}}.$$
- De plus on a :
$$h(x_0) \equiv 0 \pmod{N^m}..$$

$$\Rightarrow \quad h(x_0) = 0 \quad \text{(Lemme de Håstad/Howgrave-Graham).}$$

Résumé visuel : Théorème de Coppersmith



Résumé

- Reformulation du problème de racine mod N en problème de réseau.
- LLL produit un polynôme $h(x)$ à petits coefficients.
- Les racines entières de $h(x)$ donnent les petites racines x_0 de $f(x)$.

Application Détaillée à RSA avec $e = 3$

Application

Soient $\mathbf{C_1}$ et $\mathbf{C_2}$ **deux chiffrés** d'un même message $\mathbf{M}$, mais avec deux **bourrages** aléatoires distincts $\mathbf{r_1}$ et $\mathbf{r_2}$. Et posons $\mathbf{M_1} = \mathbf{2^m M} + \mathbf{r_1}$ et $\mathbf{M_2} = \mathbf{2^m M} + \mathbf{r_2}$ où m est la longueur du bourrage(en bits).

$\mathbf{M_1} = \mathbf{2^m M} + \mathbf{r_1}$ et $\mathbf{M_2} = \mathbf{2^m M} + \mathbf{r_2}$ où m est la longueur du bourrage(en bits).

$\mathbf{C_1} = \mathbf{M_1^3} \mod \mathbf{N}$ et $\mathbf{C_2} = \mathbf{M_2^3} \mod \mathbf{N}$.

On pose $\mathbf{\Delta} = \mathbf{r_2} - \mathbf{r_1}$ alors $\mathbf{M_2} = \mathbf{M_1} + \Delta$ et donc

$$\mathbf{C_2} = (\mathbf{M_1} + \mathbf{\Delta})^3 \mod N$$

Étape 1: Construction du Polynôme Modulaire

Technique du résultant

Utilisation de la technique du **résultant** pour éliminer la variable du message $\mathbf{M_1}$ des deux congruences polynomiales:

1. $g_1(x) = x^3 - C_1 \equiv 0 \mod N$
2. $g_2(x) = (x + \Delta)^3 - C_2 \equiv 0 \mod N$

Le résultant $h(y) = \textbf{Res}_x(g_1(x), g_2(x))$ est:

$h(\Delta) = \Delta^9 + (3C_1 - 3C_2)\Delta^6 + (3C_1^2 + 21C_1C_2 + 3C_2^2)\Delta^3 + (C_1 - C_2)^3 \equiv 0 \mod N$

Étape 2: Application du Théorème de Coppersmith

Δ la racine inconnue

1. **Polynôme** $h(\Delta)$ de degré $k = 9$.
2. **Racine** Δ (la différence des bourrage).

Puisque $\mathbf{r_1} < 2^m$ et $\mathbf{r_2} < 2^m$ et on a $\Delta = r_2 - r_1 \Rightarrow \Delta < 2^m$
D'aprés le **Théorème de Coppersmith** la(ou les) racine(s) Δ peut(peuvent) être trouvée(s) efficacement si elle(s) est(sont) inférieure(s) à $N^{1/k}$, où k est le degré du polynôme.
Dans ce cas $k = 9$, l'attaquant peut trouver Δ si:

$$|\Delta| < N^{1/9}$$

Donc la différence de bourrage Δ est récupérable si la longueur du remplissage m (où $|\Delta| < 2^m$) est inférieure à $\log_2(N^{1/9})$, c'est-à-dire moins de $1/9$ de la longueur de N.

Étape 3: Attaque de Franklin et Reiter

Relation affine

Une fois Δ trouvée, on a $\mathbf{M_2} = \mathbf{M_1} + \Delta$, ce qui est une relation affine connue. L'attaquant peut alors utiliser **l'attaque de Franklin et Reiter** pour récupérer $\mathbf{M_1}$ et $\mathbf{M_2}$, et finalement le message original $\mathbf{M}$.

Théorème de Franklin-Reiter

Soit $(\text{e}=3, N)$ une clé **RSA** publique et soient M_1 et M_2 deux éléments différents $\mathbf{Z}_\mathbf{N}^*$ de $M_2 = f(M_1) \mod \mathbf{N}$ où f est une **application affine** $f(x) := ax + b$ avec $b \neq 0$.
Alors connaissant $(\mathbf{N},\text{e}, \mathbf{C_1}, \mathbf{C_2}, f)$ on peut retrouver $\mathbf{M_1}$ et $\mathbf{M_2}$ en en temps **quadratique en** $log\mathbf{N}$.

Généralisation et Portée de l'Attaque

Généralisation

L'attaque s'applique à tout exposant public e:

1. Le polynôme résultant a un degré $k \leq e^2$.
2. La condition pour que l'attaque réussisse est que la longueur du bourrage m soit inférieure à $1/e^2$ **de la longueur** de N.
3. Pour l'exposant recommandé $e = 2^{16} + 1 = 65537$, l'attaque est impraticable car la **borne** $1/e^2$ est **trop petite**.

Références

[1] D. Coppersmith. *Finding a Small Root of a Univariate Modular Equation*. Proceedings of EUROCRYPT '96, LNCS 1070, pp. 155-165, Springer-Verlag (1996).

[2] . Franklin and M. Reiter. *A Linear Protocol Failure for RSA with Exponent Three*. Présenté à la rump session, Crypto '95. (Source citée par Coppersmith).

[3] . Howgrave-Graham. *Finding Small Roots of Univariate Modular Equations Revisited*. Cryptography and Coding, LNCS 1355, pp. 131-142, Springer-Verlag (1997).

[4] . May. *Using LLL-reduction for solving RSA and factorization problems*. Chapitre 11 de The LLL Algorithm - Survey and Applications, pp. 315-348, Springer (2009).

Merci de Votre Attention!

Questions???